张熙

技术工程部 - 基础架构团队





- 12年加入美团
- 技术工程部 基础架构中心服务治理团队负责人
 - OCTO:分布式服务通信框架及服务治理系统
 - HULK:容器集群管理及弹性伸缩平台
- 专注于面向服务架构、服务治理、大规模分布式系统、高性 能通信框架、容器化、弹性调度等领域



- 系统容量概述
- 门票业务演进历程
- 系统容量优化之道



- 场景一: PM要做一个比较大的运营促销活动, 老大过来问
 - 机器能抗住么?
 - 如果扛不住,需要加多少台机器?
- 场景二:被运维吐槽
 - 这么点调用量,还需要加机器?
 - CPU利用率怎么这么低?
- 场景三:经常收到调用方反馈
 - 你这个服务怎么一到高峰期就大量超时?



• 系统容量: 服务在保障SLA情况下能承载的极限负荷

- 怎么做容量预估?
 - 量化: 当前够不够, 多少才满足要求
- 怎么诊断容量瓶颈?
 - 寻找短板: 木桶效应
- 怎么做系统容量优化?
 - 提升效率: 尽量少的资源







景点推荐

跟团游



北京欢乐谷(4A)

★★★★ 4.7分 | 103054人消费

¥58起

票 配团 套餐

北京欢乐谷 | 15.8km



北京世界花卉大观园(4A)

★ ★ ★ ★ 4.7分 | 5854人消费

¥34.2€

满城2元 票 套督

草桥...22.3km



故宫博物院(5A)

★★★★ 4.8分 I 152640人消费

¥**5**起

票 混团 **意餐** 王府井/东单 | 12.7km



富国海底世界

★★★★ 4.2分 | 1029人消费

¥77.5起

票 混团

工体 1 9.6km



第五届北京农业嘉年华

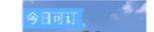
暂无评分 | 3418人消费

¥**40**起

限时減5元 票 湿团 套餐



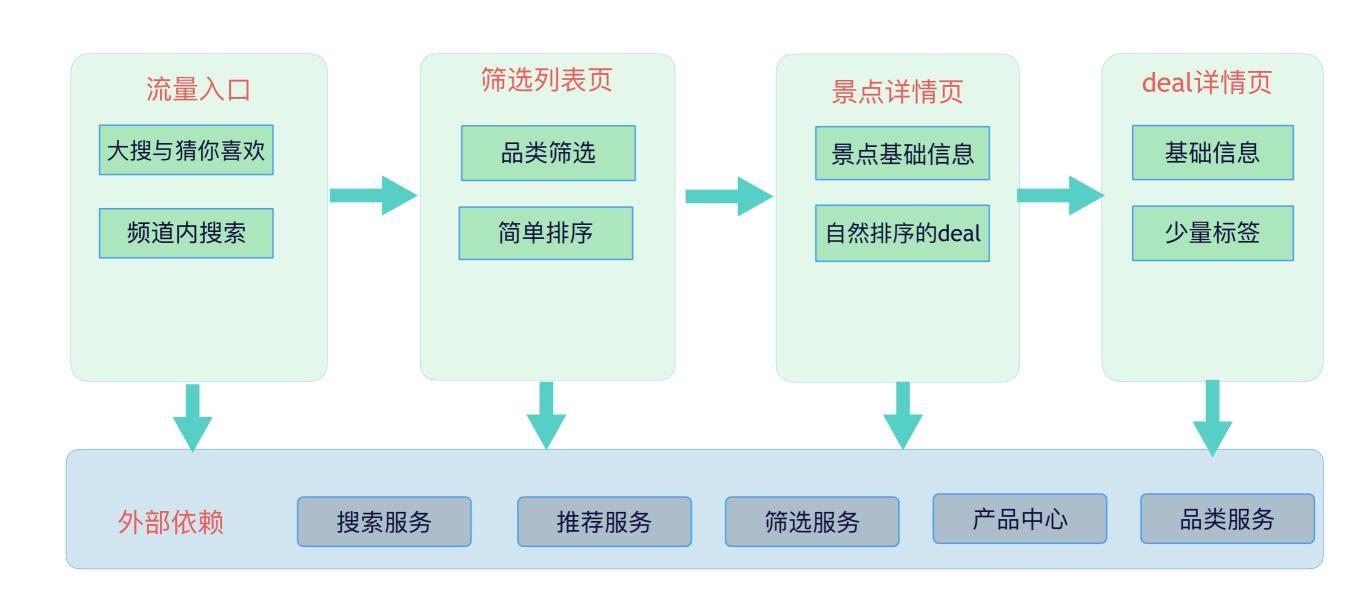




圆明园(4A)

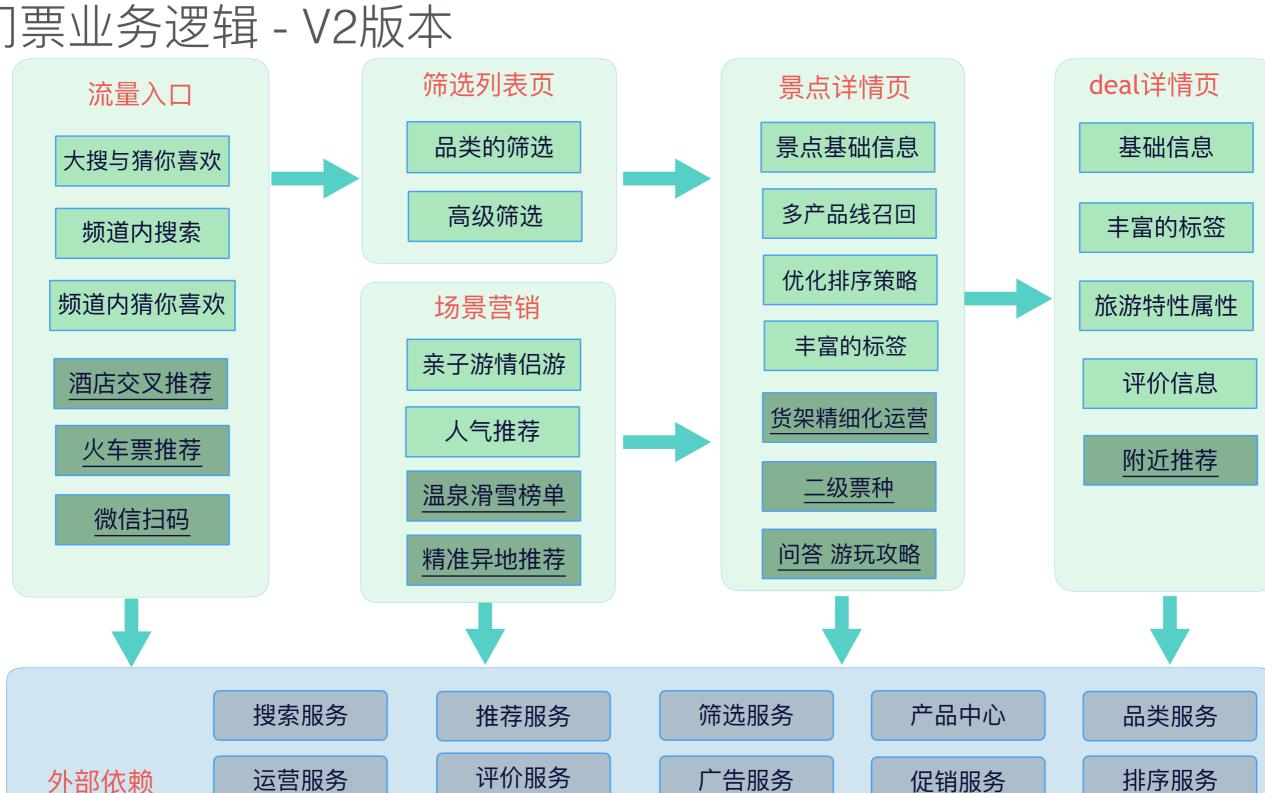


门票业务逻辑 - V1版本





门票业务逻辑 - V2版本



榜单服务

攻略服务

问答服务

票种服务



- 业务背景
 - 快速增长
 - 产品逻辑复杂
 - 外部依赖增多

- 期间如何持续保障好系统容量?
 - 评测
 - 诊断
 - 优化
 - 规划





- 评测容量
 - 单机
 - 单台服务器压测的阈值
 - 关注指标: qps、RT50/90/99、load、gc
 - 集群
 - DB限制
 - Cache限制
 - 下游依赖限制
- 评测方式
 - 引流压测:受集群实际流量限制、需高峰期、无需清理数据
 - 模拟压测:不受实际流量限制、时间灵活、适合读请求
 - 全链路压测:最准确、需要底层组件支持



事务模式

递增模式

初始用户数 10	递增用户数 10 	结束用户数	40	*QPS=并发用户数*1000/响应时间(ms	
运行时长(s) 60					
单施压机最大并发数 5	*默认100,取值范围[4,100]				
施压机数 8	*施压机数=(并发用户数-1)/单施压机最大并发数+1				
压力工具所在机房 大兴	*线上压测选大兴,线下选永丰				
● pol评情贝以略-杨刖利 ⊌童看				四戊戊戌 4	
 POIID获取POI简介-杨前利 ☑查看 			配比 4		

falcon监控

□ 是否需要与高峰时段进行对比

分组名	主机名(Falcon endpoint)	falcon环境		监控阈值	i
被压机器	dx-trip-volga09	0	cpu.idle load.1min jvm.fullgc.count	> >	3 16 5



首页 > 酒店旅游 > 旅游门票 > cambridge > 混合事务(添加CPU监控) > 构建 #268							
Console输出 测试报告 Falcon监控数据 Octo监控数据 对比报告	测试配置						
基线构建编号 260 ▼	优化前	优化后					
指标	#260	#268					
并发数	50	100.7					
单施压机最大并发数	5	5					
QPS	247	586.₹					
平均响应时间	77 ms	53 ms ³ ⊌					
top90响应时间	145 ms	108 ms ั≌					
错误率	0.00%	0.00%					
请求量	44784	141441 🗷					



• 诊断瓶颈

- 集群
 - DB:链接数、慢SQL、并发读写、主从同步、流量
 - Cache: 超时率、热点key、命中率、流量
 - 下游依赖:超时率、容量、链路分析
- 单机
 - CPU: top、jstack、JProfiler
 - 内存: jmap、MAT
 - 网络: netstat、iftop
 - 业务逻辑



• 容量优化

• DB:读写分离、分库分表、SQL优化(Atlas、Zebra、SQLAdvisor)

• Cache:序列化、mget/mput、异步

• CPU: 性能热点

• 下游依赖:链路分析、异步化

• 业务逻辑:精简、梳理



- 性能热点
 - JSON解析
 - 循环调用
 - 签名/加密
 - 日志输出

```
Thread selection: | State All thread groups
Thread status:
                               Runnable
                 83.8% – 933 s HTTP: /volga/api/v5/trip/deal/poi/full/274627.
                      83.7% - 932 s. com. meltuan.j monitor. collector. http://httpMonitorFilter.doFilter.
           ▼ 10 ===== 83.7% - 932 s org.eclipse.jetty.servlet.ServletHandler$CachedChain.doFilter
                🔻 🦲 ====== 83.7% – 932 s com.meituan.liyou.volga.app.filter.SecretFilter.doFilter
                     83.7% - 932 s org.eclipse.ietty.servlet.ServletHandler$CachedChain.doFliter
                          🔻 🦲 ====== 83.7% – 932 s. com. meituan. lvyou. volga. app. filter. FlowDemotionFilter. doFilter.
                                   📆 ===== 83.3% – 928 s. org.eclipse.jetty.servlet.ServletHandler$CachedChain.doFilter.

    83.2% – 927 s com.meituan.lyou.volga.app.filter.FlowControlFilter.doFilter.

                                          🔻 🔯 🚃 83.2% – 927 siorg.eclipse.jetty.servlet.ServletHandler$CachedChain.doFilter
                                                            83.2% - 927 s com. meituan. lvyou. volga. a pp. filter. CommonParam Filter. do Filter.
                                                    🔻 🔟 💳 83.2% – 926 s org.eclipse. Jetty. serviet. Serviet Handler $Cached Chain. do Filter
                                                         🔻 🦲 ====== 83.2% – 926 s com.meituan.trip.resilience.flowdetect.FlowdetectFilter.doFilter
                                                              $\bigsim \bigsim \b
                                                                    🔻 🔟 ===== 74.0% – 825 s com. meituan. lvyou. volga. app. controller. de al. Trip Deal List By Polid Controller. get Deals By Pol V5 Pull
                                                                        ▼ 0 ==== 74.0% - 825 s com.meituan.hyvou.volga.biz.service.deal.impl.DealMashupServiceimpl.getDealsByPolid
                                                                              🔻 🥽 ==== 56.0% – 623 s com.meituan.lvyou.volga.biz.service.deal.impl.DealFieldNewServiceimpl.getTripDealJsonArray
                                                                                   💌 🖦 💳 53.7% – 598 s. com.meituan. lyyou.yolga. biz. service. deal. impl. Deal Field New Service impl. fill Pol Detail Mp Deal ison
                                                                                        🔻 📦 💳 51.5% – 573 s. com. meituan. lvyou. volga. biz. service. deal. impl. Deal Field Util Service impl. get BuyNote
                                                                                             $\text{$\pi$} \text{$\pi$} = $1.5\% - $73 s com_meltuan_hvou_volta_biz_service_deal_impl_DealFieldUtilServiceImpl_processBuyNotes
                                                                                                  ▼ 51.3% - 57Z s com.alibaba.rasgson.jsun.tojsun
                                                                                                             🔻 🗓 💳 51.3% - 572 s com.alibaba.fastjson.jSON.tojSON
                                                                                                                  48.4% - 539 s com.alibaba.fastjson.util.TypeUtils.computeGetters

    48.4% – 539 s com.alibaba.fastjson.util.TypeUtils.computeGetters

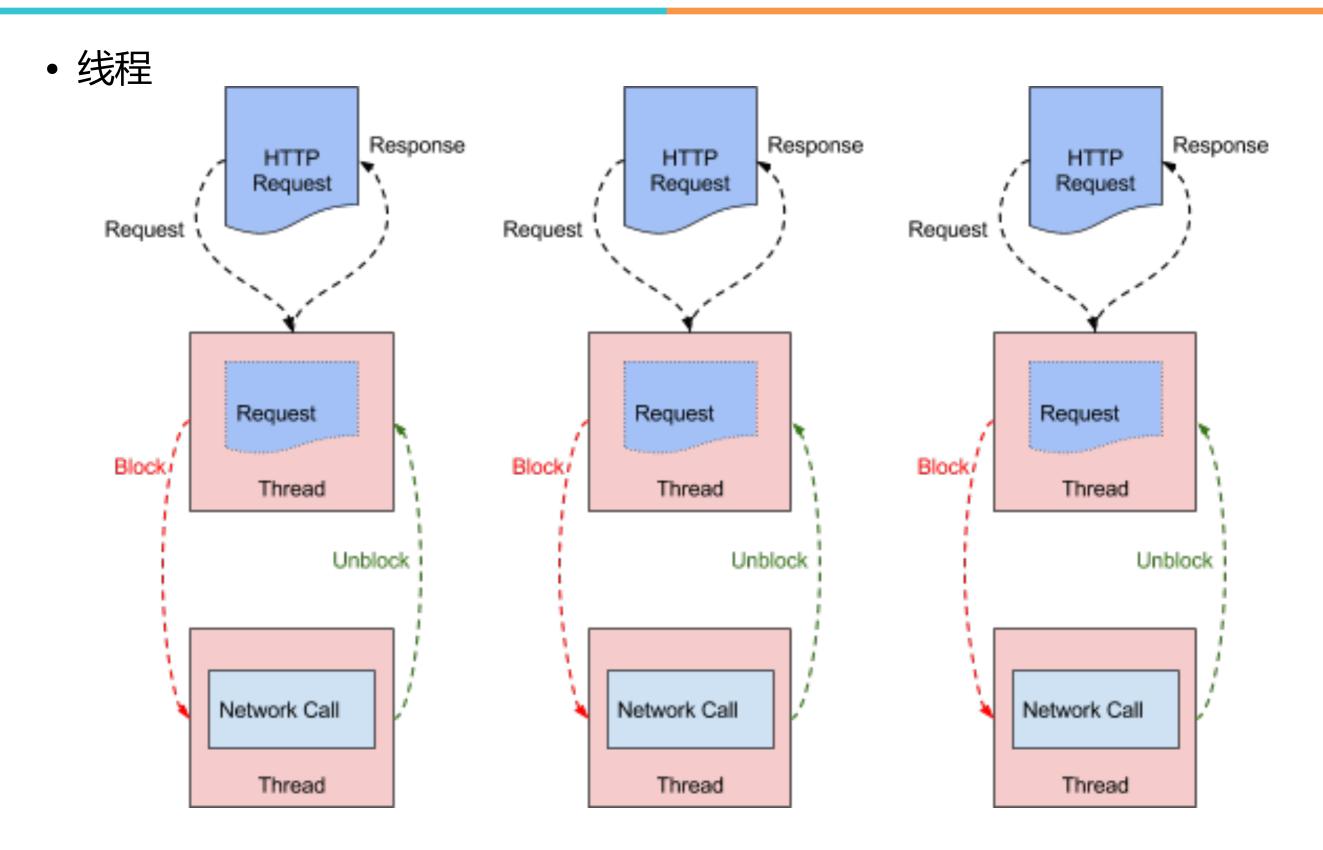
                                                                                                                            🔻 🔯 💳 46.5% – 517 s com.alibaba.fastjson.util.Fieldinfo.<init>
                                                                                                                                 46.5% - 517 s com.alibaba.fastjson.util.Fieldinfo.<init>
                                                                                                                                           😘 💳 46.4% – 517 🛊 Java. lang. reflect. Method. get Generic Return Lype 📗

    0.0% - 530 ms com.alibaba.fastjson.util. lypeUtils.setAccessible

                                                                                                                                           n 0.0% − 181 ms java.lang.reflect.Method.geti³arameterTypes
                                                                                                                                 😘 0.6% – 6,381 ms java.lang.reflect.Method.getParameterTypes.
                                                                                                                            © 0.4% – 3,907 ms com.alibaba.fastjson.util. TypeUtils.getSupperMethodAnnotation.
                                                                                                                            © 0.3% – 3,249 ms com.alibaba.fastjson.util. TypeUtils.isJSON TypeIgnore.
                                                                                                                                 0.1% – 1,392 ms java.lang.Class.getMethods.
                                                                                                                            © 0.1% = 990 ms com.alibaba.fastison.parser.ParserConfig.getField.
                                                                                                                                 0.1% – 690 ms java.lang.reflect.Method.getAnnotation.
                                                                                                                                 0.1% - 633 ms java.util.Collections.sort
                                                                                                                                 🔟 0.0% – 492 ms java.util.HashMap.put
                                                                                                                                 🕠 0.0% – 485 ms java.lang.Class.getHelds
                                                                                                                                 🕦 0.0% – 331 ms java.lang.reflect.Held.getAnnotation
                                                                                                                                 🔞 0.0% – 87,501 µs java.lang.String.startsWith
                                                                                                                                🕠 0.0% – 69.239 us javallang.StringBuilder.appendi
                                                                                                                                0.0% – 53,230 μs java.lang.Class.getAnnotation.
                                                                                                                                 🔞 0.0% – 14,847 μs java.lang.StringBuilder.toString
                                                                                                                  1.5% - 16,833 ms com.alibaba.fastison.parser.ParserConfig.isPrimitive
                                                                                                                       🔯 1.4% – 15,133 ms com.alibaba.fastjson.JSON.toJSON

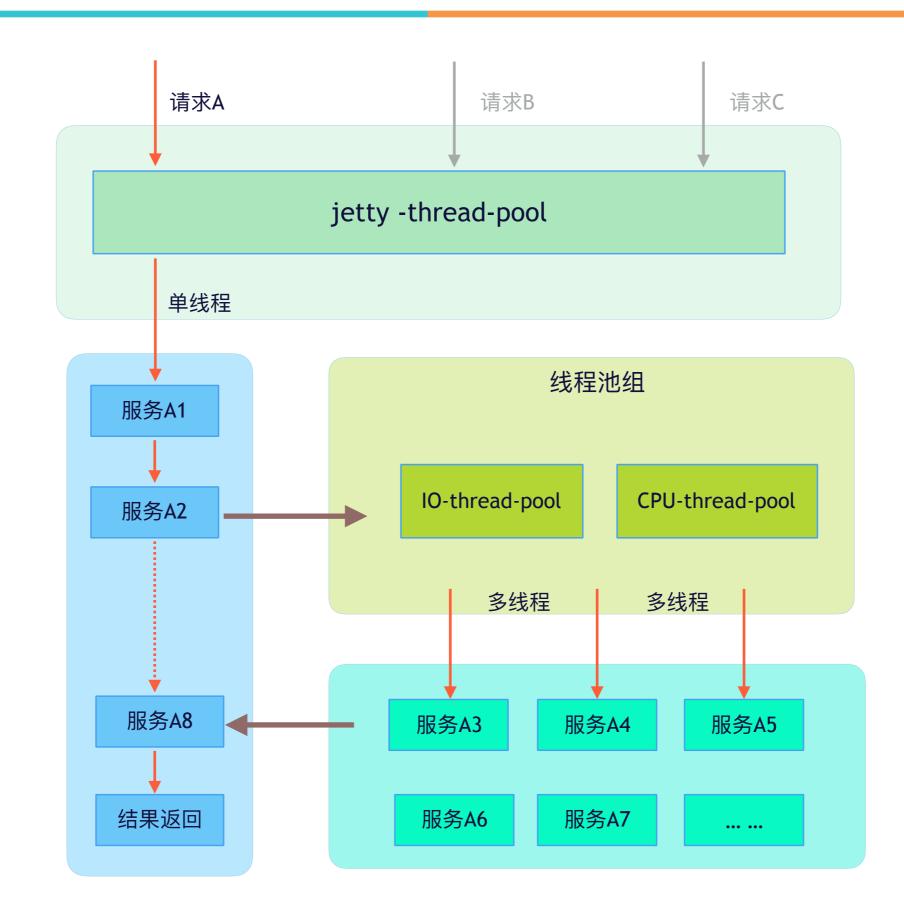
    0.0% - 546 ms com.alibaba.fastjson.util.Fieldinfo.get
```

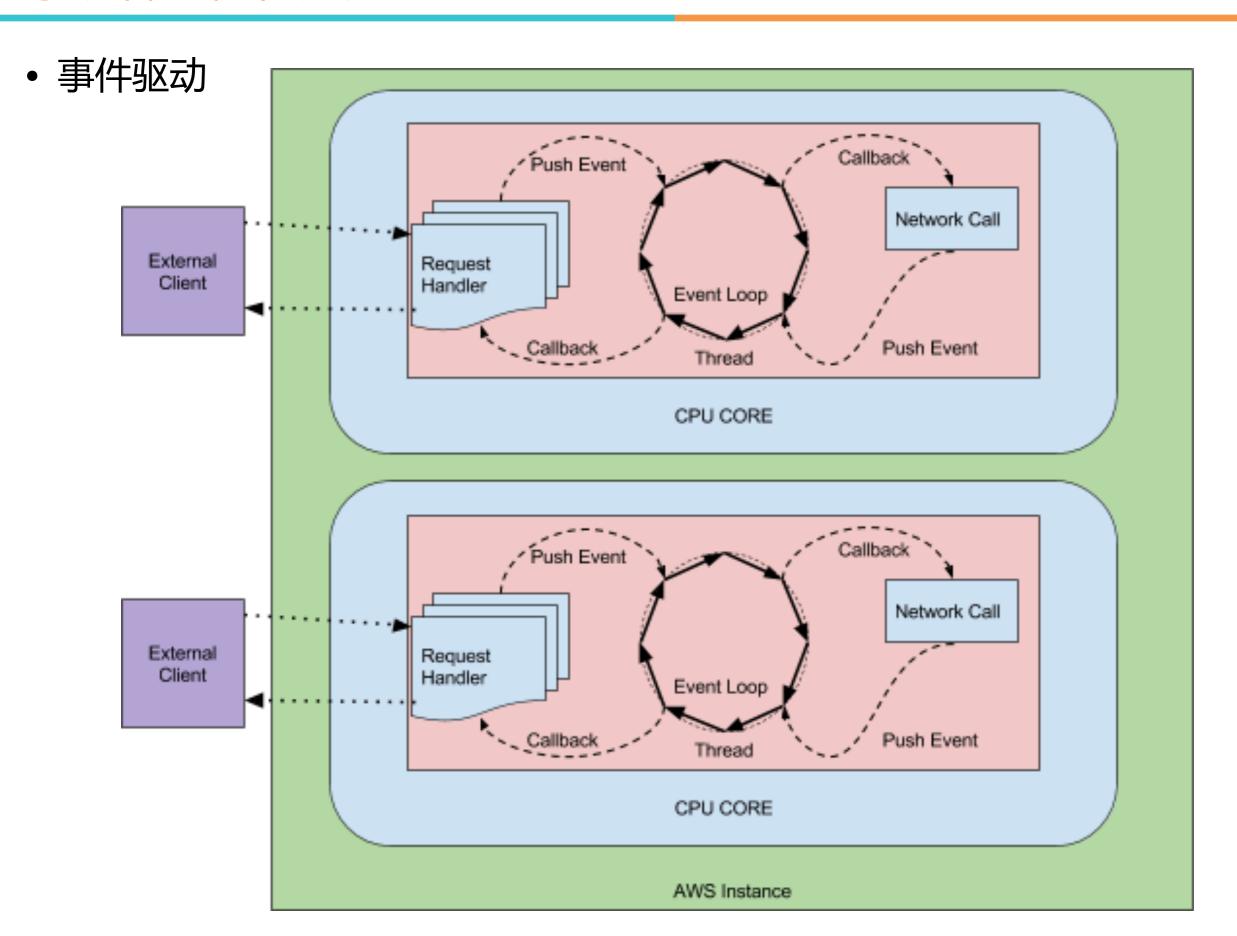






• 线程池







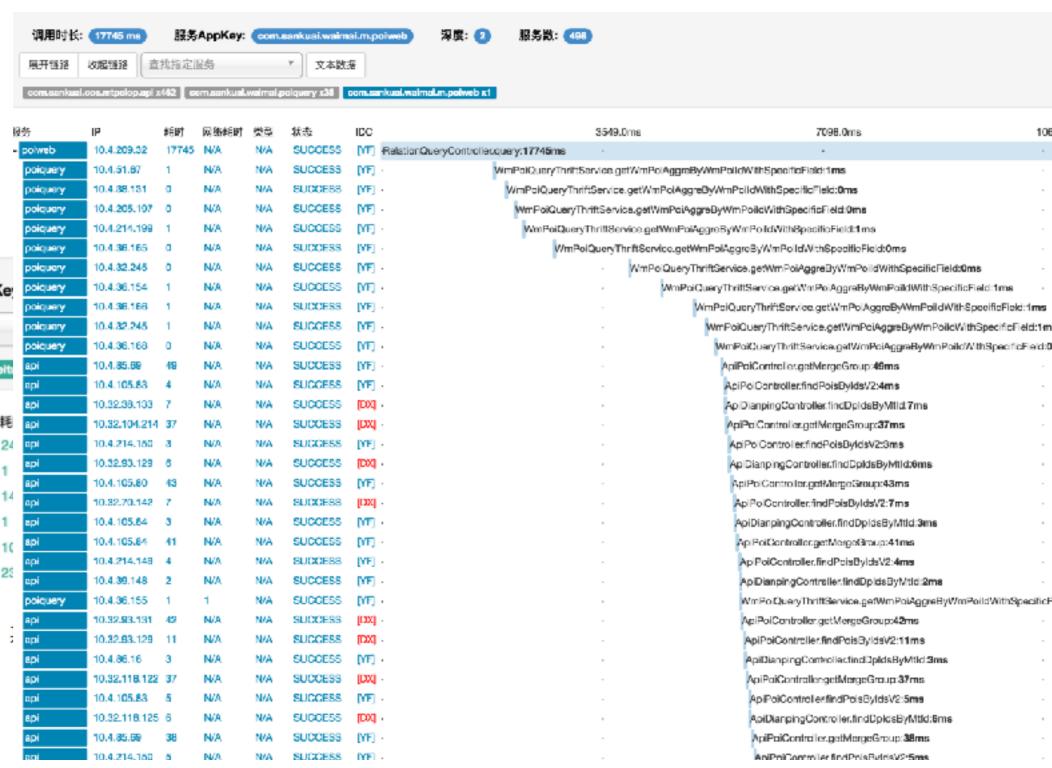
- 基础架构组件
 - RPC框架: mtthrift异步化
 - HTTP调用: Apache HttpAsyncClient
 - Cache: Cellar、Squirrel异步接口
- 异步化编程
 - Command模式
 - Java8 CompletableFuture
 - guava ListenableFuture



- 链路分析
 - 链路过长
 - 多次调用



如果一个api耗时242ms的接口,





• 多次调用

• RPC:合并请求、批量接口

• 缓存: mget、mput

• DB: 批量查询

• 业务逻辑梳理

- 不必要的代码逻辑、远程调用
- 冗余数据传输、精简数据



• 容量规划

- 单机容量 = 单台机器压测的阈值(满足SLA指标)
- 单机负荷 = 线上单台机器的峰值QPS
 - 峰值QPS估算 = (日PV数 * 80%) / (每天秒数 * 20%)
- 集群容量 = 单机容量 * 实际机器数 * 容量系数
- 容量系数 = 取决于集群依赖资源的容量,如DB、Cache、下游服务
- 容量倍数 = 集群容量 / (实际机器数 * 单机负荷)

• 机房容量

- 单机房水位:1.25倍容量、1.5倍容量、2倍容量
- 双机房水位: 2倍容量、2.5倍、3倍容量



• 系统容量

- 集群:机器数、DB、Cache、网络带宽
- 单机:资源瓶颈、同步调用、业务逻辑

• 容量优化之道

- 评测:引流、模拟、全链路
- 诊断: DB、Cache、下游依赖、CPU、业务逻辑
- 优化:性能热点、异步化、链路分析、逻辑梳理
- 规划
 - QPS估算 = (日PV数 * 80%) / (每天秒数 * 20%)
 - 容量倍数 = 集群容量 / (实际机器数 * 单机负荷)





Thank You!



Email: zhangxi@meituan.com

微博: @lispmind

